**שאלה 2**

1. : האלגוריתם שלנו יתבסס על כך שנצמיד לעץ הגבוהה יותר את העץ הנמוך יותר, וההצמדה תתבצע בצומת פנימית בעץ הגבוהה יותר. ידוע שכל מפתח ב- קטן מכל מפתח ב-. נניח בה"כ . נתחיל לרדת מהשורש של כלפי שמאל (נרד תמיד שמאלה בעץ, לכיוון הערך המינימלי בעץ), ובסה"כ נעבור על קשתות ונגיע לצומת כלשהו ב-T2 שנקרא לו - (במקרה שבו , כשמספר הצעדים יוצא שלילי, נייצר צומת חדש ששמו והאינדקס שלו יהיה , כלומר הערך של המפתח המינימלי ב-. לאחר מכן, נעביר את להיות תת העץ הימני של צומת זה, ונשאיר את תת העץ השמאלי שלו להיות – בהמשך תת העץ השמאלי שלו יוגדר כמו שצריך). ידוע לנו מהו ערך המינימלי מבין כל המפתחות ב-, ונסמנו ב-. **טיול בעץ (ואולי יצירה של צומת חדש) מהשורש אל מתבצע בסיבוכיות זמן ומקום .** מכיוון שעברנו על קשתות בסה"כ, הרי שהגובה של הצומת הוא  
   כי הנחנו . כעת יכולים להיות מספר אפשרויות עבור הצומת , ונפתור כל מקרה בנפרד:

* **מקרה 1:** הצומת זהו הצומת שיצרנו בתחילת התרגיל, במקרה שבו . במקרה זה, כעת בצומת זה קיים המפתח , ותת העץ הימני שלו הוא שגובהו . נגיד כתת העץ השמאלי של את העץ שגובהו הוא גם . דבר זה אפשרי מכיוון שכל המפתחות ב- קטנים מכל המפתחות ב-, ובפרט גם מהמפתח של שערכו . סה"כ להעביר מספר קבוע של מצביעים, ויצירה של צומת חדש **לוקח סיבוכיות זמן וסיבוכיות מקום.** בסופו של דבר נקבל עץ חדש שגובהו .
* **מקרה 2:** בצומת קיים אינדקס יחיד שנסמנו , כלומר רק תת עץ שמאלי ותת עץ ימני (ללא תת עץ מרכזי). נסמן את תת העץ הימני ב- ואת תת העץ השמאלי . גובה הצומת הוא ולכן גובה 2 תתי העצים הוא . נוסיף לצומת אינדקס נוסף כך ש-. נניח בשלילה שמתקיים . לכן, כי הערך המינימלי בעץ. מכאן נובע ש- הוא הצומת בעל המפתח המינימלי בעץ, ולכן הוא בתחתית העץ, ולכן עברנו בסה"כ קשתות, אבל וזו סתירה (כי עברנו בסה"כ קשתות, וברור ש- כי במקרה זה אין סיבה לבצע , כלומר זהו מקרה מנוון). לכן חייב להתקיים ש-. כמובן שזה הגיוני כי הרי הערך המינימלי בעץ () נמצא בתת העץ השמאלי של צומת . כעת, נוכל למקם את העץ כתת עץ שמאלי בצומת החדש מכיוון שידוע שמתקיים , והרי הוא האינדקס השמאלי בצומת . לכן, נמקם את העץ שגובהו כתת עץ שמאלי ב-, את תת העץ כתת עץ מרכזי ב- (שגובהו גם ), ואת תת העץ כתת עץ ימני ב- (שגובהו גם ). נציין כי לא הרסנו את גובהו של העץ כי הגובה של הצומת הוא , ולכן הוספה של תת עץ שמאלי בגובה תשמור על שמורה זו. בסה"כ ביצענו כמות סופית של החלפת מצביעים והשמות, **ולכן פעולה זו תתבצע בסיבוכיות זמן ובסיבוכיות מקום כי לא ביצענו הקצאות דינמיות.**
* **מקרה 2:** בצומת קיימים 2 אינדקסים, , ומכאן שלצומת זה יש תת עץ שמאלי שנסמנו , שבו כל המפתחות בתחום , תת עץ מרכזי שנסמנו , שבו כל המפתחות בתחום ותת עץ ימני שנסמנו , שבו כל המפתחות בתחום . בדומה למקרה הקודם, נוסיף אינדקס לצומת כך שהתת עץ השמאלי יהיה (שמקיים שכל המפתחות בו בתחום וגובהו . תת העץ השני משמאל יהיה , תת העץ השלישי משמאל יהיה , ותת העץ הימני יהיה .

לפני:



אחרי:



גובה כל תתי העצים הוא , ולכן גובה הצומת נותר . נפצל את הצומת ל-2 צמתים . הצומת יכיל את המפתח . תת העץ השמאלי שלו יהיה , ותת העץ הימני יהיה . הצומת יכיל את המפתח . תת העץ השמאלי שלו יהיה ותת העץ הימני שלו יהיה . שינוי ועדכון מספר קבוע של מצביעים, וביצוע מספר קבוע של השמות ופיצולים מתבצע בסיבוכיות זמן ומקום .כעת, נחזור במעלה הרקורסיה לכיוון השורש תוך כדי תיקון העץ בדיוק לפי אלגוריתם הכנסת איבר בעץ 2,3 שלמדנו בכיתה (שמטפל בפיצולים). **סיבוכיות הזמן והמקום לחזרה לשורש תהיה**  מכיוון שגובה העץ החדש יהיה לכל היותר , מכיוון שהצומת נמצא בעומק , נצטרך לבצע לכל היותר פיצולים, השמות והעברת מצביעים בדרך חזרה אל השורש. מכיוון שתיקון הצומת מתבצע ב-, **נקבל שסה"כ סיבוכיות הזמן והמקום היא .** במקרה זה, מכיוון שהיינו צריכים לפצל צומת, גובה העץ החדש יהיה .

1. נבצע בסיבוכיות לפי סעיף א'.
2. נשתמש באינדוקציה:

**בסיס:** נאחד את העצים בעזרת . נסמן את העץ החדש ב- . סיבוכיות הזמן והמקום היא לפי סעיף 1. גובה העץ החדש יהיה לפחות ולכל היותר (לפי סעיף 1).

**הנחה:** נניח שהאיחוד של עצים (על ידי ביצוע , ואז ביצוע , וכו'...) מתבצע בסיבוכיות זמן ומקום . נקרא לעץ המאוחד . בכל איחוד גובה העץ גדל ב-1 לכל היותר (לפי סעיף 1). לכן עבור איחודים גובה העץ יגדל ב- לכל היותר. נסיק שגובה העץ יהיה לפחות ולכל היותר . כלומר

**צעד:** נבצע איחוד ל- (גם על ידי ביצוע סדרה של לפי סדר העצים). לפי הנחת האינדוקציה, סיבוכיות הזמן והמקום לאיחוד העצים (נסמן את סדרת האיחודים הזאת ב-) היא . לכן קיים כך שלכל מתקיים: . כעת, סיבוכיות הזמן והמקום לביצוע האיחוד האחרון (בין העץ לבין העץ ) היא (נסמן את האיחוד האחרון ב-). לכן קיים כך שלכל מתקיים:

נבחר ואז נקבל שלכל :  
.ידוע שגובה העץ הוא לפחות ולכל היותר . לכן נחלק למקרים:

* **אם** אז , ולכן:

**ולכן סיימנו עבור מקרה זה**.

* **אחרת, אם**  אז על פי הנתון ש- ועל פי העובדה ש-, נקבל:

ולכן:

***ולכן גם סיימנו עבור מקרה זה.***

**לכן, עבור עצים לאיחוד נקבל שסיבוכיות הזמן והמקום היא , כנדרש.**

**שאלה 4**

**חלק ראשון**

נשתמש במבנה הנתונים Trie שלמדנו בקורס, כאשר כל צומת יהיה מהצורה:



*בנוסף, נשמור משתנה נוסף בצד ששמו .*

**הגדרה:** 2 מוצרים ייקראו **חופפים** אם תיאור 2 המוצרים מתחיל באותה תת מחרוזת כלשהי. לדוגמא המחרוזות ו- הן חופפות, אך המחרוזות ו- אינן חופפות.

*נסביר את המשמעות של כל אחד מהאיברים שהצומת מחזיק:*

* *: אם הצומת הוא הצומת האחרון שמייצג את המוצר בעץ (הצומת שמגיע לאחר $), ערך זה יכיל את מחיר המוצר. אחרת, ערך זה יכיל את המחיר המינימלי מבין כל המוצרים החופפים שצומת זה מייצג. לדוגמא עבור העץ המכיל את המחרוזות abcd, abce, abcf, הצומת המגיע לאחר הקשת c יכיל ב- את המחיר המינימלי מבין כל שלושת המוצרים. דוגמא עבור מצב בסיסי במוצר ‘ab’:*



* : מכיל את המחרוזת של המוצר הקודם שהוכנס למערכת בזמן שבו הוכנסה המחרוזת הנוכחית. הצומת היחיד שמכיל בעל משמעות הוא הצומת האחרון שאליו אנו מגיעים לאחר (יוסבר בהמשך באופן יותר מפורט כיצד מעדכנים את איבר זה).
* : מערך באורך שמחזיק מצביעים לבנים (כפי שלמדנו בתרגול).
* : רשימה מקושרת של המצב הקודם של כל ה- בכל הצמתים שנמצאים במסלול של המוצר בגרף. לרשימה זו יש משמעות רק בעלים של העץ, כי היא מייצגת את המצב הקודם של ה- בכל המסלול (מהשורש אל העלה). בכל אחת מהצמתים הפנימיים של העץ, ערך משתנה זה תמיד יהיה .

**נתאר כעת את מהלך הפעולות במבנה:**

* **:** נאתחל עץ ידי יצירת ששמו , ולאחר מכן בנייה של מערך הבנים כך שכל האיברים במערך יצביעו ל-. בנוסף, נבצע השמה: , , ו- .  **סיבוכיות הזמן והמקום היא**  מכיוון שביצענו כמות סופית של פעולות (גם הקצאת המערך ואתחולו מתבצעים ב- כי גודל המערך קבוע), וביצעו הקצאת זיכרון סופית ולא דינמית.



* **:** נאתחל רשימה מקושרת . נתחיל לעבור על המסלול בעץ ה- המתאר את המחרוזת , ונייצר את הבנים הרלוונטיים במהלך הדרך. בכל בן שנייצר, נאתחל את משתניו להיות זהים למשתנים שאתחלנו בצומת בפונקציה . עבור כל צומת שנעבור בו בדרך (כולל צומת שזה עתה ייצרנו – אם בכלל יצרנו - ולא כולל הצומת האחרון במסלול), נוסיף את הערך היושב ב- בצומת לראש הרשימה . לאחר מכן, נעדכן את הערך היושב ב- להיות המינימלי מבין ובין . לבסוף, בצומת האחרון שנייצר שמגיע לאחר הקשת $, נעדכן , כלומר נשמור ב- את מחיר המוצר שעתה הכנסנו, , כלומר נשמור ב- את שם המוצר שנכנס לפני המוצר שזה עתה הכנסנו, ונשים את הרשימה המקושרת שייצרנו ב- (עדכונים אלו מתבצעים בצומת האחרון בלבד). לאחר מכן נבצע: , כלומר נשמור ב- את שם המוצר שכעת הכנסנו. סיבוכיות הזמן לייצר כל אחד מהצמתים בדרך, להוסיף איברים לרשימה המקושרת, ולעדכן את כל המשתנים בתוך כל צומת היא כי עבור כל צומת אנו מבצעים כמות קבועה של פעולות. אנו מייצרים בסה"כ צמתים ( צמתים עבור המחרוזת ועוד צומת אחד עבור התו ), ומבצעים מספר קבוע של השוואות והשמות בין משתנים בכל צעד ברקורסיה (שעומקה ), **ולכן סיבוכיות הזמן והמקום בסה"כ הם .**
* **:** אם אינו מכיל דבר (None), זה אומר שעוד לא הכנסנו אף מחרוזת למבנה הנתונים, ולכן לא נבצע דבר. אחרת, נייצר משתנה חדש בשם , ונעביר אליו את המחרוזת הנמצאת ב-. על פי הגדרת המבנה שלנו, כל צומת שהוא סיומת של מחרוזת של מוצר כלשהו (צומת שמגיע לאחר ) מכיל בתוכו את שם המוצר שנכנס לפניו, את מחיר המוצר , ורשימה מקושרת של המצב הקודם לכל המשתנים הנמצאים בצמתים על המסלול של המוצר. נעבור על המסלול שמייצג את המחרוזת בעץ ה- שלנו. לבסוף, כשנגיע לצומת האחרון במסלול (הצומת שלאחר הקשת ), נשלוף מצומת זה את ערך המשתנה ונבצע השמה של מחרוזת זו אל תוך . כלומר, נעדכן את כך שיכיל את שם המוצר לפני המוצר שאנו מבצעים לו . בנוסף, נשלוף את הרשימה המקושרת מהצומת האחרון ונשמור אותה בצד לשימוש עתידי. נחזור אחורה במסלול, ובכל צומת שנעבור בו נשלוף את המספר הבא מ-. אם לצומת אין בנים, זה אומר שהצומת רלוונטית רק במסלול של המוצר שאנו מוציאים, ולכן נמחק את הצומת זה. אחרת, אם לצומת יש בנים, זה אומר שיש מוצר שחופף למוצר שאנו מוציאים, ולכן נעדכן את בצומת זה להיות הערך שזה עתה שלפנו מהרשימה המקושרת . מספר קבוע של השמות והעברת ערכים ממשתנה אחד לשני מתבצעת בסיבוכיות זמן . בנוסף, ביצענו מעבר רקורסיבי על מסלול בעץ ה- באורך כדי להגיע לצומת האחרון שמייצג את המוצר שאנו מבצעים לו . מעבר זה כמובן מתבצע בסיבוכיות זמן ומקום . לבסוף, מעבר חזרה על אותו מסלול בגרף שבמצע את המחיקה של המוצר יתבצע גם כן בסיבוכיות זמן , אך בסיבוכיות מקום כי אנו מבצעים לכל היותר מחיקות, כמות קבועה של השמות והעתקות משתנים, ולא מבצעים שום הקצאה נוספת. **לסיכום, סיבוכיות הזמן והמקום היא כנדרש.**
* **:** נעבור רקורסיבית על המסלול בעץ ה- שמייצג את המוצר ששמו עד אשר נגיע לצומת האחרון במסלול. נשלוף את הערך שנמצא במשתנה בצומת האחרון במסלול, ונחזיר את ערך זה למשתמש במערכת. סה"כ ביצענו מעבר רקורסיב על מסלול באורך , ולכן המעבר על כל הצמתים והגעה לצומת האחרון במסלול מתבצע בסיבוכיות זמן ומקום . שליפה של ערך ממשתנה מתבצעת בסיבוכיות זמן , וללא הקצאות נוספות, **בסה"כ נקבל שסיבוכיות הזמן והמקום היא .**

**חלק שני**

המבנה שבנינו בחלק הראשון כבר מתאים לתמיכה בפעולה הנוספת של החלק השני, ועומד בכל דרישות הסיבוכיות גם של החלק הראשון וגם של החלק השני. נתאר את מהלך הפעולה החדשה שהתווספה:

* **:** נניח ש-. נעבור על המסלול בגרף : במידה ועבור צומת כלשהו לא קיימת קשת (), זה אומר שהמוצר לא קיים במערכת ונחזיר . במקרה זה נעבור לכל היותר על קשתות ולכן נקבל סיבוכיות זמן . אחרת, אם הצלחנו לעשות מעבר על הקשת האחרונה , בצומת שהגענו אליו קיים כבר הערך הרלוונטי להחזרה. על פי הגדרת מבנה הנתונים שלנו, המשתנה בצומת שנמצא לאחר הקשת מכיל את המחיר המינימלי מבין כל המוצרים החופפים שצומת זה מייצג. לדוגמא, עבור העץ הבא ו-:



הצומת שלאחר הקשת מכיל את המחיר 100, שהוא

המחיר המינימלי מבין המוצרים ab ו-ac.

לכן, נחזיר למשתמש את הערך הנמצא במשתנה בצומת

שלאחר הקשת . **סה"כ עברנו על צמתים ברקורסיה, ולכן סיבוכיות הזמן והמקום**

**של פעולה זו תהיה .**

**שאלה 5**

נשתמש ב-2 מחסניות LIFO לצורך המימוש של מחסנית ה-FIFO. להלן מחסנית MAIN ומחסנית QUEUE (הסימון Bottom מסמל את תחתית המחסניות):



**InitQ():** נבנה 2 מחסניות LIFO ששמן MAIN ו-QUEUE על ידי קריאה ל-InitStack() פעמיים. סיבוכיות הזמן במקרה הגרוע היא O(1) כי ביצענו בסה"כ 2 פעולות שנתון שסיבוכיות הזמן שלהן במקרה הגרוע היא O(1), ולכן אנו עומדים בסיבוכיות הזמן הנדרשת.

**EnQ(x):** נבצע Push(x) למחסנית MAIN. סיבוכיות הזמן במקרה הגרוע עבור הכנסה אחת היא O(1) (נתון).

**DeQ(x):** נבדוק האם המחסנית QUEUE ריקה על ידי קריאה לפונקציה IsEmpty() בסיבוכיות זמן O(1) במקרה הגרוע (נתון). **אם QUEUE ריקה**, נעתיק את המחסנית MAIN אל המחסנית QUEUE באופן הבא: כל עוד המחסנית MAIN אינה ריקה, נבצע Pop() לאיבר x מהמחסנית MAIN ואז נכניס אותו למחסנית QUEUE על ידי שימוש ב-Push(x). נקרא לפעולה זו מעתה **העתקה הפוכה של MAIN ל-QUEUE:**



ברור כי כשנסיים לבצע את פעולת ההעתקה ההפוכה, נקבל שהמחסנית MAIN תהיה ריקה והמחסנית QUEUE תכיל את כל האיברים שהיו ב-MAIN בסדר הפוך מהסדר שבו הם הוכנסו ל-MAIN מלכתחילה (האיבר שהוכנס ראשון ב-MAIN יהיה האיבר בראש המחסנית QUEUE, והאיבר שהוכנס אחרון ב-MAIN יהיה בתחתית המחסנית QUEUE). אם לפני פעולת ההעתקה ההפוכה קיימים m איברים במחסנית MAIN, נראה שסיבוכיות הזמן לבצע העתקה הפוכה של MAIN ל-QUEUE היא O(m) במקרה הגרוע: אנו מבצעים כמות סופית של פעולות בעת העתקה של כל איבר מ-MAIN ל-QUEUE, כאשר כל פעולה לוקחת O(1) במקרה הגרוע (אנו מבצעים Push, Pop, ו-IsEmpty עבור כל איבר). מכיוון שיש בסה"כ m איברים ב-MAIN, אנו מבצעים בסה"כ m פעולות בסיבוכיות של O(1) במקרה הגרוע, ולכן סיבוכיות הזמן הכוללת של ההעתקה ההפוכה היא O(m) במקרה הגרוע. כעת, לאחר ההעתקה, נבצע Pop() לאיבר הנמצא בראש המחסנית QUEUE, ובכך קיבלנו את האיבר הראשון שהכנסנו למחסנית MAIN ולכן בסה"כ קיבלנו מימוש Pop של מחסנית FIFO בסיבוכיות זמן O(m) במקרה הגרוע. **אחרת, אם QUEUE אינה ריקה**, זה אומר שביצענו בעבר העתקה הפוכה של MAIN ל-QUEUE (כי אין דרך אחרת שבה QUEUE יכולה להתמלא ע"פ מבנה הנתונים שהגדרנו), ולכן האיבר הראשון שהוכנס למחסנית MAIN נמצא בראש מחסנית QUEUE ע"פ מה שהסברנו קודם (גם אם MAIN המשיכה להתמלא, עדיין QUEUE מכילה את האיברים הראשונים שהוכנסו ל-MAIN). לכן אם נבצע Pop() למחסנית QUEUE נקבל כמו מקודם את האיבר הראשון שהוכנס למחסנית MAIN, אך בשונה ממקודם, מכיוון ש-QUEUE כבר מלאה ולא היינו צריכים להעתיק אליה את MAIN, ומכיוון שהשתמשנו פעם אחת בלבד ב-Pop() ופעם אחת בלבד ב-IsEmpty() כדי לבדוק אם QUEUE אינה ריקה, נקבל שסיבוכיות הזמן במקרה הגרוע להוצאת האיבר היא O(1). **לסיכום: אם QUEUE ריקה, סיבוכיות הזמן להוצאת האיבר הראשון שהוכנס ל-MAIN היא O(m) במקרה הגרוע כאשר m מסמל את כמות האיברים ב-MAIN, ואם QUEUE אינה ריקה, סיבוכיות הזמן להוצאת האיבר הראשון שהוכנס ל-MAIN היא O(1) במקרה הגרוע.**

**כעת, ננתח את הסיבוכיות המשוערכת של הפעולות EnQ(x), DeQ(x).**

**הוכחה בעזרת שיטת הצבירה:** תהי סדרה של m פעולות מהסוגים EnQ(x) ו-DeQ(x). ונסמן את סדרת הפעולות ב-, כאשר היא הפעולה הראשונה בסדרה. **נחלק את סדרת הפעולות ל-** **תתי סדרות** באופן הבא: נסרוק את סדרת הפעולות לפי הסדר (מהפעולה הראשונה לאחרונה). נייצר תת סדרה חדשה. כל עוד הפעולה הנוכחית היא EnQ(x), נוסיף את הפעולה אל תת הסדרה שייצרנו. אחרת (אם הפעולה הנוכחית היא DeQ(x)), נוסיף אותה לתת הסדרה שייצרנו, ונייצר תת סדרה חדשה שאליה נוסיף את הפעולות הבאות שנמצא בסריקה. נקבל תתי סדרות כאשר כל תת סדרה של פעולות היא מהצורה: EnQ(x), EnQ(x), … DeQ(x). במידה ולא היה קיים אף DeQ(x), נקבל תת סדרה אחת שמכילה רק EnQ(x) (שגם תהיה שווה לסדרת הפעולות המקורית). **במקרה זה** (שלא קיים DeQ(x)), מכיוון שאנו מבצעים m פעולות EnQ(x), ומכיוון שכל פעולה מבוצעת בסיבוכיות זמן של O(1) במקרה הגרוע, נקבל שסיבוכיות הזמן עבור m פעולות היא O(m), ולכן סיבוכיות הזמן המשוערכת היא O(1). **אחרת**, אם קיימת פעולת DeQ(x) אחת לפחות, ננתח את סיבוכיות זמן הריצה במקרה המשוערך עבור תת סדרת הפעולות ה- (): נניח ובתת סדרת הפעולות ה- קיימות פעולות בסה"כ. ידוע כי הפעולות הראשונות יהיו EnQ(x) (ככה מוגדרת תת הסדרה), ומכיוון שכל פעולת EnQ(x) לוקחת O(1) סיבוכיות זמן במקרה הגרוע, נקבל שסיבוכיות הזמן במקרה הגרוע היא . כעת הפעולה ה- היא בטוח DeQ(x) (על פי הגדרת תת סדרת הפעולות) וראינו שבמקרה הגרוע היא עולה לנו (יש בסה"כ במחסנית MAIN מכיוון שלפני שביצענו את תת סדרת הפעולות הנוכחית: או שלא ביצענו כלום והתור היה ריק (נתון), או שביצענו DeQ(x) מתישהו שרוקנה את MAIN). כלומר עבור סדרת הפעולות ה- שבה יש פעולות, קיבלנו שסיבוכיות הזמן במקרה הגרוע היא . מכיוון שבסה"כ יש פעולות, ברור כי . לכן, סיבוכיות זמן הריצה של כל תתי הסדרות ביחד תהיה סיבוכיות זמן הריצה של סדרת הפעולות המקורית (כי תתי הסדרות הן בעצם פירוק של סדרת הפעולות המקורית לחלקים נפרדים). סיבוכיות זמן הריצה של כל תתי הסדרות היא במקרה הגרוע, ולכן סיבוכיות זמן הריצה המשוערכת של EnQ(x) ו-DeQ(x) היא במקרה הגרוע.

מ.ש.ל.

***הוכחה בעזרת שיטת החיובים:*** *נגדיר תשלומים באופן הבא:*

* *: המחיר המשוערך של* (כל איבר שנכנס ל-MAIN חייב לצאת דרך QUEUE, ולכן נשלם עבור push אל MAIN, לאחר מכן pop מ-MAIN, לאחר מכן push למחסנית QUEUE, ואז pop נוסף במידה ונוציא את איבר זה בעתיד).
* *: המחיר המשוערך של .*

**עבור**  המחיר בפועל הוא (בדיוק כמו Push() רגיל). את היתרה (3) נצמיד לאיבר . המחיר בפועל של **הפעולה**  תלויה בהאם ריקה או לא. **אם היא ריקה**, אז המחיר בפועל יהיה כאשר זה כמות האיברים הנוכחית ב-MAIN: לכל אחד מאיברים אלו יש יתרה של של 3. ניקח מכל איבר יתרה של 2 עבור ההוצאה מ-MAIN וההכנסה ל-QUEUE (מכאן בא ה-). בנוסף ניקח את היתרה 1 שנותרה לאיבר שאנו בפועל מוציאים. כלומר, נשלם את פעולה זו במקרה זה על ידי שימוש ביתרה שהותרנו בכל אחד מהאיברים. **אחרת, אם לא ריקה**, המחיר בפועל הוא שזהו היתרה שנותרה לאיבר שאנו מוציאים מראש המחסנית (כל איבר שעובר מ-MAIN ל-QUEUE נותר עם יתרה של 1). לכן נשלם את פעולה זו בעזרת היתרה שנותרה. בשני המקרים, לא נשלם על פעולה זו כלל, כי יש לנו מספיק יתרה בבנק. נקבל:

ניתן לחסום את זמן הריצה על ידי:

מ.ש.ל.

**הוכחה בעזרת שיטת הפוטנציאל:** נגדיר את פוטנציאל המבנה: כאשר זה מספר האיברים במחסנית לאחר פעולות. נראה שהמחיר המשוערך חסום על ידי קבוע עבור שני פעולות המבנה:

* : . כלומר, לאחר הפעולה שעולה לנו בפועל , הגדלנו את הפוטנציאל ב-3 כי הוספנו איבר אחד למחסנית .
* : . נבצע חלוקה למקרים: **אם ריקה,** מתקיים ש-

במקרה זה מכיוון שזאת העלות בפועל שתעלה לנו להעביר את האיברים מ- אל , ואז להוציא את האיבר שבראש . בנוסף כי לאחר פעולה זו, מחסנית מתרוקנת (ככה מוגדר המבנה כאשר ריקה).

**אם אינה ריקה**, מתקיים ש-

כלומר יעלה לנו רק להוציא את האיבר מ-, ובכך גם לא שינינו את הפוטנציאל כי כמות האיברים ב- נשארת זהה. מכיוון ש- נקבל:

מ.ש.ל.